E-Companion for "GradPlanner: A Deployed Multi-Objective Optimization–Based Decision Support Tool for Student Degree Planning and Timely Graduation"

EC-I. Multi-objective optimization solution method: detailed description

We now present the mixed-integer linear programming formulation of the SGPP. Using the notation summarized in Tables EC-1–EC-3, we solve the SGPP with a staged procedure that separates feasibility, graduation time, and plan-quality objectives. Stage 0 (underload control) relaxes minimum-credit requirements only as much as necessary: Stage 0.a minimizes the number of underloaded terms, and Stage 0.b, with that count fixed, minimizes the worst underload amount, producing a policy-consistent underload profile $(Z^\star, \varepsilon^\star)$. Stage 1 then fixes this profile and minimizes the makespan T, yielding the shortest contiguous graduation horizon compatible with prerequisites, offerings, and credit limits. Finally, Stage 2 keeps both the underload profile and makespan fixed and refines the plan with respect to the similarity and difficulty balance penalties: short lexicographic runs compute attainable ranges for each penalty, and a Chebyshev max–min problem in normalized utility space selects a Pareto-optimal plan that balances alignment to the 4-year reference plan with an even distribution of course difficulty across terms.

Constraints

Given the notation in Tables EC-1–EC-3, the SGPP is governed by the following constraints. Before presenting them formally, we briefly summarize their role. The model assigns each remaining course to exactly one future term, restricts courses to terms in which they are offered, enforces prerequisite order, and keeps each used term within the allowed credit range. Additional constraints identify which terms are used, prevent gaps in enrollment, and allow limited underload only when needed to preserve feasibility. Together, these constraints translate the main advising rules into a mathematical planning model.

Single Course Assignment Constraint Each course must be scheduled in exactly one term.

$$\sum_{t=1}^{\mathcal{S}} x_{c,t} = 1 \qquad \forall c \in C \tag{1}$$

Course Availability Constraint A course can only be scheduled in a term in which it is offered.

$$x_{c,t} \le A_{c,t} \qquad \forall c \in C,\ \forall t \in \mathcal{S} \tag{2}$$

Table EC-1 Sets and indices

Symbol	Definition
C	The finite set of courses in the program.
$C^{\text{taken}} \subseteq C$	The set of courses that have already been completed.
$C^{\text{must}} \subseteq C$	The set of courses that must be scheduled.
$\mathcal{S} = \{1, \ldots, S\}$	The set of semesters (indexed by t).
$\mathcal{P} \subseteq C \times C$	The set of prerequisite arcs, where $(i \to j)$ indicates that course i must be completed before course j.
c	Course index $(c = 1, 2, \ldots, C)$.
t	Semester index $(t = 1, 2, \ldots, S)$.
i, j	Course indices used to denote prerequisite arcs.

Table EC-2 Model parameters

Parameter	Definition
R_c	Reference term for course c.
$D_{c,t} = \lvert t - R_c \rvert$	Precomputed distance between the scheduled term t and the reference term for course c.
Q_c	Credit hours of course c.
W_c	Difficulty weight of course c.
$H_t^{\min}$	Minimum credit-hour capacity for term t.
$H_t^{\max}$	Maximum credit-hour capacity for term t.
$A_{c,t} \in \{0, 1\}$	Indicator: $A_{c,t} = 1$ if course c is offered in term t; otherwise $A_{c,t} = 0$.
M	A sufficiently large positive constant (big-M).
ρ	Chebyshev tie-breaker weight.
$\Delta_{\text{simpenalty}}$	Normalization range for similarity penalty: $\text{SIMPenalty}_{\max} - \text{SIMPenalty}_{\min}$.
$\Delta_{\text{balpenalty}}$	Normalization range for difficulty-balance penalty: $\text{BALPenalty}_{\max} - \text{BALPenalty}_{\min}$.

Table EC-3 Decision variables

Variable	Definition
$x_{c,t} \in \{0, 1\}$	$x_{c,t} = 1$ if course c is scheduled in term t; otherwise $x_{c,t} = 0$.
T	Makespan of the schedule.
$y_t \in \{0, 1\}$	$y_t = 1$ if term t contains at least one scheduled course; otherwise $y_t = 0$.
l_t	Difficulty load of term t.
μ	Average difficulty load across used terms.
e_t	Abs-value auxiliaries for difficulty balance penalty value.
$z_t \in \{0, 1\}$	Underload indicator in term t (1 if underload occurs; otherwise 0).
v_t	Slack variable representing the underload amount in term t.
ε	Small tolerance used in Stage 0.b.
$u_{\text{simpenalty}} \in [0, 1]$	Normalized utility for similarity penalty.
$u_{\text{balpenalty}} \in [0, 1]$	Normalized utility for difficulty balance penalty.
$\tau \in [0, 1]$	Chebyshev max–min utility level.

Course dependency constraint The next constraint encodes the main sequencing logic of degree planning: a prerequisite course i must be completed in an earlier term before its successor course j can be scheduled.

$$\sum_{k=1}^{t} x_{j,k} \;\le\; \sum_{k=1}^{t-1} x_{i,k} \qquad \forall (i,j) \in \mathcal{P},\ \forall t \in \mathcal{S}. \tag{3}$$

Term usage links The following constraints identify which terms are active and prevent unrealistic plans with empty semesters between used terms. A term can be marked as used/ active only if at least one course is scheduled in that term.

$$y_t \le \sum_{c \in C} x_{c,t} \qquad \forall t \in \mathcal{S} \tag{4a}$$

$$x_{c,t} \le y_t \qquad \forall c \in C,\ \forall t \in \mathcal{S} \tag{4b}$$

No empty term in the middle Once terms become unused, all later terms must also be unused, so there are no gaps in the schedule.

$$y_t \ge y_{t+1}, \qquad \forall\, t \in \{1, \ldots, S-1\} \tag{5}$$

Makespan linearization The makespan T is defined as the total number of used terms.

$$T = \sum_{t=1}^{S} y_t \tag{6a}$$

$$1 \le T \le S \tag{6b}$$

The credit-load constraints ensure that each used term respects the institution's maximum-load policy, while the minimum-load requirement is relaxed only through controlled underload variables when strict enforcement would make a realistic student case infeasible.

Maximum Term capacity constraint This constraint ensures that the total credit hours scheduled in any term cannot exceed that term's maximum credit capacity.

$$\sum_{c \in C} Q_c\, x_{c,t} \;\le\; H_t^{max} \qquad \forall t \in \mathcal{S}. \tag{7}$$

Minimum Term Capacity Constraint The minimum-load requirement is relaxed only through controlled underload variables, captured by v_t, when strict enforcement would make a realistic student case infeasible.

$$\sum_{c \in C} Q_c\, x_{c,t} \;\ge\; H_t^{min} y_t - v_t \qquad \forall t \in \mathcal{S} \tag{8}$$

Link underload slack (v_t) to underload indicator (z_t) and term usage indicator (y_t) The underload slack (v_t) can be positive only if the term is used and flagged as underloaded.

$$v_t \le M\, z_t \qquad \forall t \in \mathcal{S}, \tag{9a}$$

$$v_t \le M\, y_t \qquad \forall t \in \mathcal{S}. \tag{9b}$$

Per-term underload cap The underload in any term cannot exceed the global tolerance ε.

$$v_t \le \varepsilon \qquad \forall t \in \mathcal{S} \tag{10}$$

Variable bounds in Stage 0 The underload slack and its tolerance are constrained to be nonnegative.

$$v_t \ge 0 \qquad \forall t \in \mathcal{S}, \tag{11a}$$

$$\varepsilon \ge 0. \tag{11b}$$

Difficulty Load of a term The difficulty load of a term is the sum of the difficulty weights of all courses scheduled in that term.

$$l_t = \sum_{c \in C} W_c\, x_{c,t} \qquad \forall t \in \mathcal{S} \tag{12a}$$

$$l_t \ge 0 \qquad \forall t \in \mathcal{S} \tag{12b}$$

Average difficulty load of schedule The average difficulty load μ is defined as the mean term difficulty over the used terms.

$$\mu \;=\; \frac{1}{T^*} \sum_{t \in \mathcal{S}} l_t \tag{13a}$$

$$\mu \ge 0 \tag{13b}$$

Stage 0: Underload control

Departmental policies typically require a minimum credit load per used term (e.g., full-time status, financial aid, visa, or progress-to-degree rules). In practice, meeting this minimum can be infeasible in certain semesters due to course availability, prerequisite chains, or previously taken courses

that remove feasible combinations. Therefore, we do not impose the minimum credit load as a hard constraint. Instead, before the main stages, we handle this soft constraint through a two-step lexicographic optimization: 0.a minimize the number of underloaded terms using binary flags z_t; then 0.b with that count fixed, minimize the severity of underload by bounding the per-term slack v_t with a common scalar ε. This yields schedules that respect institutional intent as much as possible without rendering the model infeasible.

Stage 0.a: lexicographic optimization, minimize the number of underloaded terms We first minimize the number of underloaded terms (terms below the minimum credit), $\sum_{t\in\mathcal{S}} z_t$ where $z_t \in \{0,1\}$ flags underload and slack $v_t \geq 0$ captures its amount.

$$Z^* := \min \sum_{t\in\mathcal{S}} z_t \tag{14}$$
$$\text{s.t. (1)-(11a),}$$

Stage 0.b: Fix the z_t and minimize the underloaded amount With $\sum_{t\in\mathcal{S}} z_t$ fixed at its optimum Z^*, minimize the worst underload across counted terms via a scalar $\varepsilon \geq 0$, yielding the smallest possible severity consistent with Stage 0.a.

$$\varepsilon^* := \min \varepsilon \tag{15}$$
$$\text{s.t. (1)-(11b),}$$
$$\sum_{t\in\mathcal{S}} z_t = Z^\star, \tag{15-fix}$$

Stage 1: Minimize Makespan

This stage selects the shortest contiguous schedule horizon. Contiguity is enforced so no empty term appears before a used term, and the makespan T counts the number of used terms. The objective is then:

$$T^* := \min T \tag{16}$$
$$\text{s.t. (1)-(11b), (15-fix),}$$
$$\varepsilon = \varepsilon^\star \tag{16-fix}$$

Stage 2: balanced multi-objective optimization

Similarity penalty This parameter penalizes deviations from the department's reference 4-year plan by measuring how far each scheduled course is placed from its reference term. Let $R_c \in \{1, \ldots, S\}$ denote the reference term for course c, and define $D_{c,t} = |t - R_c|$ as the absolute deviation if course c is taken in term t. The distances $D_{c,t}$ are precomputed for all $c \in C$ and $t \in \mathcal{S}$. The total similarity penalty is:

$$\text{SIMPenalty} = \sum_{c \in C} \sum_{t \in \mathcal{S}} D_{c,t}\, x_{c,t} \tag{17}$$

Difficulty balance penalty Advising practice seeks to spread workload so students do not cluster many hard courses in the same term, but exact equalization is often infeasible due to offerings, prerequisites, and taken courses. Thus we penalize imbalance, rather than enforce. Let l_t be the difficulty of term t and μ the Average Difficulty of the schedule. We introduce gated absolute deviations e_t (activated only when $y_t = 1$ via M link). The total balance penalty is calculated as:

$$e_t \geq l_t - \mu - M\,(1 - y_t) \qquad \forall t \in \mathcal{S}, \tag{18a}$$

$$e_t \geq \mu - l_t - M\,(1 - y_t) \qquad \forall t \in \mathcal{S}, \tag{18b}$$

$$e_t \leq M\, y_t \qquad \forall t \in \mathcal{S}, \tag{18c}$$

$$e_t \geq 0 \qquad \forall t \in \mathcal{S}, \tag{18d}$$

$$\text{BALPenalty} = \sum_{t \in \mathcal{S}} e_t \tag{18e}$$

Lexicographic ranges for normalization After fixing schedule feasibility (Stage 0) and schedule makespan (Stage 1), we quantify two secondary losses, SIMPenalty and BALPenalty. We then obtain lexicographic bounds for each loss so that they can be normalized and combined in a balanced way via Chebyshev scalarization. Specifically, we compute $\text{BALPenalty}_{\min}$, $\text{BALPenalty}_{\max}$, $\text{SIMPenalty}_{\min}$, and $\text{SIMPenalty}_{\max}$ via four short solves.

a. Minimize $BALPenalty$ to get $BALPenalty_{min}$ We find the best (lowest) difficulty balance penalty so we can later normalize. Let $\text{BALPenalty}_{\min}$ denote the optimal objective value of (19a).

$$\text{BALPenalty}_{\min} := \min\ \text{BALPenalty} \tag{19a}$$

$$\text{s.t. (1)-(13b), (15-fix), (16-fix), (17), (18a)-(18e),}$$

$$T = T^{\star} \tag{19a-fix}$$

b. Minimize $SIMPenalty$ at fixed $BALPenalty = BALPenalty_{min}$ to get $SIMPenalty_{max}$ With difficulty balance penalty fixed at its best, we then find the worst compatible similarity penalty to form the upper bound for similarity penalty. Let $\text{SIMPenalty}_{\text{max}}$ denote the optimal objective value of (19b).

$$\begin{aligned} &\text{SIMPenalty}_{\text{max}} := \min \text{ SIMPenalty} && \text{(19b)}\\ &\text{s.t. (1)-(13b), (15-fix), (16-fix), (17),} \\ &\text{(18a)-(18e),(19a-fix),} \\ &\text{BALPenalty} = \text{BALPenalty}_{\text{min}} && \text{(19b-fix)} \end{aligned}$$

c. Minimize $SIMPenalty$ to get $SIMPenalty_{min}$ We find the best (lowest) similarity penalty so we can later normalize. Let $\text{SIMPenalty}_{\text{min}}$ denote the optimal objective value of (19c).

$$\begin{aligned} &\text{SIMPenalty}_{\text{min}} := \min \text{ SIMPenalty} && \text{(19c)}\\ &\text{s.t. (1)-(13b), (15-fix), (16-fix), (17),} \\ &\text{(18a)-(18e)(19a-fix)} \end{aligned}$$

d. Minimize $BALPenalty$ at fixed $SIMPenalty = SIMPenalty_{min}$ to get $BALPenalty_{max}$ With similarity penalty fixed at its best, we then find the worst compatible difficulty balance penalty to form the upper bound for difficulty balance penalty. Let $\text{BALPenalty}_{\text{max}}$ denote the optimal objective value of (19d).

$$\begin{aligned} &\text{BALPenalty}_{\text{max}} := \min \text{ BALPenalty} && \text{(19d)}\\ &\text{s.t. (1)-(13b), (15-fix), (16-fix), (17),} \\ &\text{(18a)-(18e),(19a-fix),} \\ &\text{SIMPenalty} = \text{SIMPenalty}_{\text{min}} && \text{(19d-fix)} \end{aligned}$$

Chebyshev max–min over normalized utilities To combine similarity penalty and difficulty balance penalty in a balanced way, we optimize the worst normalized improvement (utility) across the two losses. Using the lexicographic bounds ($\text{SIMPenalty}_{\text{min}}$, $\text{SIMPenalty}_{\text{max}}$) and ($\text{BALPenalty}_{\text{min}}$, $\text{BALPenalty}_{\text{max}}$), we normalize each loss to a utility in $[0, 1]$ and then maximize

the common level τ (with a small tie–breaker). We apply a max–min Chebyshev scalarization in utility space. Normalize each loss using its lexicographic range by letting $\Delta_{\text{simpenalty}}$ denote normalization range for similarity penalty and $\Delta_{\text{balpenalty}}$ denote normalization range for difficulty balance penalty:

$$\Delta_{\text{simpenalty}} = (\text{SIMPenalty}_{\max} - \text{SIMPenalty}_{\min}) \tag{20a}$$

$$\Delta_{\text{balpenalty}} = (\text{BALPenalty}_{\max} - \text{BALPenalty}_{\min}) \tag{20b}$$

We also introduce $u_{\text{simpenalty}}, u_{\text{balpenalty}} \in [0,1]$ as the normalized utilities for similarity and difficulty balance, respectively, and a common utility level $\tau \in [0,1]$:

$$u_{\text{simpenalty}} = \frac{\text{SIMPenalty}_{\max} - \text{SIMPenalty}}{\Delta_{\text{simpenalty}}} \tag{20c}$$

$$u_{\text{balpenalty}} = \frac{\text{BALPenalty}_{\max} - \text{BALPenalty}}{\Delta_{\text{balpenalty}}} \tag{20d}$$

$$u_{\text{simpenalty}} \geq \tau, \qquad u_{\text{balpenalty}} \geq \tau \tag{20e}$$

Tie-broken max–min objective:

$$\begin{aligned} &\max \ \tau + \rho\,(u_{\text{simpenalty}} + u_{\text{balpenalty}}) \\ &\text{s.t. (1)-(13b), (15-fix), (16-fix), (17),} \\ &\text{(18a)-(18e), (19a-fix),} \\ &\text{(20a)-(20e)} \end{aligned} \tag{21}$$

EC-II. Algorithms

The staged GradPlanner procedure is summarized in Algorithms EC-1 and EC-2. Algorithm EC-1 presents the overall workflow used to construct and solve an SGPP instance: it removes completed courses, builds the remaining requirement set and constraints, applies the Stage 0 underload-control procedure, minimizes the graduation horizon, and then calls the secondary-objective refinement step. Algorithm EC-2 details this refinement step, where the similarity and difficulty-balance objectives are normalized and combined through the Chebyshev scalarization to select a balanced plan among the minimum-time feasible schedules. Together, the two algorithms describe the implementation logic used by the deployed GradPlanner.

Algorithm EC-1 Lexicographic–Chebyshev Optimization Pipeline for Graduation Planning

Require: $C, C^{\text{taken}}, \mathcal{S}, \mathcal{P}$; R_c, Q_c, W_c, $\mathrm{A}_{c,t}$, $\mathrm{H}_t^{\min}$, $\mathrm{H}_t^{\max}$, M, ρ

Ensure: $x_{c,t}$, T, SIMPenalty, BALPenalty,

Preprocess: remove C^{taken} from C and $\mathcal{P}$

Build base MILP: assignment, availability, capacity (min/max), prerequisite, contiguity; define $l_t = \sum_c \mathrm{W}_c\, x_{c,t}$

Stage 0.a:

Introduce $v_t \geq 0$, $z_t \in \{0, 1\}$

Solve $\min \sum_{t \in \mathcal{S}} z_t$; link v_t to y_t and z_t

Fix $\sum_t z_t = Z^\star$

Stage 0b:

Add scalar $\varepsilon \geq 0$ and constraints $v_t \leq \varepsilon$ for all t

Solve $\min \varepsilon$ and fix ε

Stage 1:

Define $T = \sum_{t \in \mathcal{S}} y_t$

Solve $\min T$ and fix T^*

Stage 2 setup:

Precompute $D_{c,t} = |t - R_c|$

SIMPenalty $\leftarrow \sum_{c,t} D_{c,t} x_{c,t}$

$\mu \leftarrow \left(\sum_t l_t\right)/T^*$

Gate absolute deviations with $y_t \in \{0, 1\}$: $\mathrm{e}_t \geq l_t - \mu - M(1 - y_t)$, $\mathrm{e}_t \geq \mu - l_t - M(1 - y_t)$, $\mathrm{e}_t \leq M y_t$

BALPenalty $\leftarrow \sum_t \mathrm{e}_t$

Stage 2 ranges; Short solves for normalization:

Find $\text{SIMPenalty}_{\min}$, $\text{SIMPenalty}_{\max}$, $\text{BALPenalty}_{\min}$, and $\text{BALPenalty}_{\max}$ via four brief lexicographic runs

Chebyshev solve:

$\langle u_{\text{simpenalty}}, u_{\text{balpenalty}}, \tau \rangle \leftarrow$ ChebyshevMaxMin()

Solve maximize $\tau + \rho\,(u_{\text{simpenalty}} + u_{\text{balpenalty}})$

return $x_{c,t}, T$, SIMPenalty, BALPenalty

Algorithm EC-2 Chebyshev Max–Min Utility Scalarization

Require: SIMPenalty, BALPenalty; [$\text{SIMPenalty}_{\min}$,$\text{SIMPenalty}_{\max}$] and [$\text{BALPenalty}_{\min}$, $\text{BALPenalty}_{\max}$]; $\rho > 0$

Ensure: $u_{\text{simpenalty}}, u_{\text{balpenalty}} \in [0,1]$, $\tau \in [0,1]$

1: $\Delta_{\text{simpenalty}} \leftarrow (\text{SIMPenalty}_{\max} - \text{SIMPenalty}_{\min})$

2: $\Delta_{\text{balpenalty}} \leftarrow (\text{BALPenalty}_{\max} - \text{BALPenalty}_{\min})$

3: Introduce $u_{\text{simpenalty}}, u_{\text{balpenalty}}, \tau \in [0,1]$

4: $u_{\text{simpenalty}} \leftarrow (\text{SIMPenalty}_{\max} - \text{SIMPenalty})/\Delta_{\text{simpenalty}}$

5: $u_{\text{balpenalty}} \leftarrow (\text{BALPenalty}_{\max} - \text{BALPenalty})/\Delta_{\text{balpenalty}}$

6: **return** $u_{\text{simpenalty}}, u_{\text{balpenalty}}, \tau$

EC-III. Benchmarking against alternative solution methods

To strengthen the validation of the proposed Chebyshev planner, we also benchmarked it against two alternative methods that were considered before adopting the final Chebyshev formulation. The first baseline is a weighted-sum MILP (with equal weights), which uses the same core SGPP constraints and the same primary graduation-time objective, but combines the two secondary objectives, SIMPenalty and BALPenalty, through an additive weighted objective. The second baseline is a rule-based heuristic that first removes completed courses from the remaining requirement set and then fills available plan slots by prioritizing courses with the largest number of dependent courses, while applying credit-load adjustments to construct feasible schedules.

Table EC-4 reports the percentage change of each baseline relative to the Chebyshev planner across the same validation instances. The weighted-sum method matches Chebyshev on average graduation time and slightly improves SIMPenalty, but its loss in BALPenalty is larger than its gain in SIMPenalty, indicating a less balanced trade-off between the two secondary objectives. The heuristic is computationally simple, but it does not guarantee optimality; in our results, it increases graduation time and more than doubles BALPenalty. These results support the use of the Chebyshev scalarization because it better balances similarity and difficulty among minimum-time plans.

EC-IV. Course attributes for the CS program

This section summarizes the course attributes of the Computer Science program in Table EC-5. The difficulty levels used in the model were initially obtained via a questionnaire completed by undergraduate advisors in the Bellini College. Advisors were given the list of required courses for the five programs in the Bellini College and asked to rate each course on a four-point scale from

Table EC-4 Performance degradation of alternative optimization methods relative to the Chebyshev planner

Method	ΔT	ΔSIMPenalty	ΔBALPenalty
Weighted-sum	0.00%	-1.24%	+2.74%
Heuristic	+2.56%	-1.24%	+107.71%

Percentage changes are computed relative to the Chebyshev planner. Positive values indicate worse performance than Chebyshev, while negative values indicate better performance.

1 (lowest) to 4 (highest). The ratings reflected both perceived workload and advising experience, including whether the course usually has more complex or technical material, whether it tends to create difficulty for students, and how often students fail, withdraw from, or repeat the course.

To reduce the concern that these weights rely only on subjective judgment, we performed an additional empirical validation using historical grade-distribution data from USF InfoCenter (University of South Florida 2026). For each required course in the CS curriculum, we retrieved grade distributions from Spring 2023 through Fall 2025 and computed weighted pass rates using the minimum grade required for progression in the program. For courses requiring a minimum grade of B, only A and B grades were counted as successful completions; for courses requiring a minimum grade of C, A, B, and C grades were counted as successful completions. For General Education, general elective, and major elective course categories, which do not correspond to a single fixed course, we selected the three most broadly used representative courses from each corresponding category and computed category-level weighted pass rates by aggregating successful completions and total attempts across those courses. The results showed consistency between advisor judgment and empirical completion risk. Table EC-6 shows the pass-rate-based empirical risk scale used for this validation, while Table EC-5 reports the advisor-informed difficulty weights used in the optimization model.

EC-V. Detailed description of synthetic input classes for solution validation

This section provides complete descriptions of the ten experimental input classes summarized in Table 2 in Section 6.2, to support reproducibility. The corresponding input files for all classes are also available in our GitHub repository.

Input Class 1: Fresh start baseline

This class represents a typical on-track baseline in which the student has taken at most 0-5 General courses, and has completed none of the CS/ math gateway courses. As a result, essentially the full four-year reference plan remains available, with all Math and CS core courses still to be scheduled.

Table EC-5 Course codes, titles, credit hours, and assigned difficulty levels

Course Code	Course Title	Credit	Difficulty
MAC2311	Engineering Calculus I	4	4
MAC2312	Engineering Calculus II	4	4
COT3100	Introduction to Discrete Structures	3	4
PHY2048/Lab	General Physics I/Lab	4	4
PHY2049/Lab	General Physics II/Lab	4	4
COP2510	Programming Concepts	3	4
CDA3103	Computer Organization	3	4
COP3514	Program Design	3	4
CDA3201/Lab	Computer Logic and Design/Lab	4	4
COP4530	Data Structures	3	4
CDA4205/Lab	Computer Architecture/Lab	4	4
COP4600	Operating Systems	3	4
CNT4419	Secure Coding	3	4
CEN4020	Software Engineering	3	4
COT4400	Analysis of Algorithms	3	4
EGN2440	Probability and Statistics with Calculus	3	3
TechElec	Technical Elective	3	3
TheoElec	Theory Elective	3	3
SoftElec	Software Elective	3	3
GenElec	General Elective	3	3
GenNat	Gen-Ed Natural Science	3	3
EGN4450	Introduction to Linear Systems	2	2
ENC1101	Composition I	3	2
ENC1102	Composition II	3	2
GenSoc	Gen-Ed Social Science	3	2
GenHum	Gen-Ed Humanities (SGEH)	3	2
CIS4250	Ethical Issues and Professional Conduct	3	1

Table EC-6 Pass-rate-based empirical risk scale used for validation

Weighted Pass Rate	Empirical Risk Level
$\geq 99.00\%$	1
95.00%–98.99%	2
90.00%–94.99%	3
$< 89.99\%$	4

This configuration is used to capture reference behavior and to generate a broad Pareto frontier: the similarity objective can closely mimic the reference plan, while the difficulty-balance objective can substantially reshape the schedule, allowing Chebyshev scalarization to operate.

Input Class 2: General Education-heavy, core-empty

This class represents a student who has completed 8-12 General Education and elective courses, but has taken no math or CS core courses. As a result, the remaining plan consists almost entirely of Math and CS core courses that must be compressed into the remaining terms. This creates strong

pressure on the similarity objective, since many early General Education slots in the reference plan are already filled, and makes difficulty balancing more challenging in later semesters, as the optimizer must schedule a dense block of core courses while staying as close as possible to the reference plan.

Input Class 3: Math-ready, CS-not-ready

This class represents a student who has taken most of the math courses, but not yet prepared for CS core courses. The taken set includes key Math courses such as MAC2311 and MAC2312, while introductory programming courses are explicitly excluded and all CS core courses remain to be taken. This configuration shifts the main bottlenecks: many math-dependent courses become available earlier, narrowing the range of feasible similarity to the reference plan, while difficulty balance stress tends to appear later as CS core courses are compressed into the remaining terms.

Input Class 4: CS-ready, Math-not-ready

This class is the symmetric counterpart of Input Class 3 and represents a student who has taken most of the CS core courses, but not yet prepared for math courses. The taken set includes introductory programming courses such as COP2510, while math gateways remain incomplete. As a result, several CS core courses can be scheduled earlier, while long prerequisite chains in math postpone many theory and upper-level courses.

Input Class 5: On-track, gateways complete

This class represents an on-track student who has completed approximately 25–30% of the degree, including all major gateway courses. The taken set includes 5–10 of the earlier courses in the plan, while none of the mid-level core courses have been taken. As a result, the remaining plan is a dense block of core and upper-level courses. Similarity to the reference plan can still be good, but the difficulty balance objective becomes challenging because many core courses must be packed into the remaining terms.

Input Class 6: On-track, mid-program

This class represents an on-track student who has completed approximately 50–55% of the degree, with all gateway courses and a substantial portion of the early and mid-level core courses already taken. The remaining plan consists primarily of upper-level CS requirements and a smaller set of electives spread over the remaining terms. Compared to earlier on-track classes, the feasible range for similarity to the reference plan is narrower, since many early slots are fixed by completed courses, while difficulty balancing remains challenging because a concentrated set of upper-level courses must still be scheduled.

Input Class 7: On-track, near-completion

This class represents an on-track student who has completed approximately 75–80% of the degree. Most gateway, early, and mid-level core courses are already finished, and several upper-level requirements may also be completed, leaving only a relatively small set of remaining upper-level courses and electives to be scheduled over the final terms. At this stage, the feasible range for similarity is quite narrow, while difficulty balancing is limited because only a few remaining courses can be rearranged in the last semesters.

Input Class 8: Near-graduation, few courses remaining

This class represents a student who has fewer than eight courses remaining to graduate. Almost all gateway, core, and required courses have already been completed, leaving only a small set of upper-level requirements, electives, and/or general education courses to be scheduled in the final terms. Planning flexibility is minimal: the plan can only deviate slightly from the reference, and difficulty balancing is limited because only a few remaining courses can be rearranged.

Input Class 9: Only low-difficulty courses completed

This class represents a student who has completed only low-difficulty courses (such as basic general education and elective courses), while most medium- and high-difficulty core and upper-level courses remain. As a result, the remaining plan is mostly harder courses, so the optimizer has little freedom to spread difficulty across terms and balancing becomes much more challenging.

Input Class 10: Only high-difficulty courses completed

This class represents a student who has already completed most high-difficulty gateway, theory, and upper-level courses, while the remaining requirements are mainly lower-difficulty general education and elective courses. The remaining plan is therefore dominated by easier courses, so difficulty balancing is less of a concern and the optimizer has more freedom to align the schedule with the reference plan.